(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 2 630 565 B1

(12) **EUROPEAN PATENT SPECIFICATION**

(45) Date of publication and mention of the grant of the patent:
12.12.2018 Bulletin 2018/50

(21) Application number: **11779511.2**

(22) Date of filing: **19.10.2011**

(51) Int Cl.:
G06F 3/06 *(2006.01)* ***G06F 13/38*** *(2006.01)*
G06F 13/40 *(2006.01)*

(86) International application number:
PCT/US2011/056850

(87) International publication number:
WO 2012/054578 (26.04.2012 Gazette 2012/17)

(54) **USB-TO-SATA HIGH-SPEED BRIDGE**

HOCHGESCHWINDIGKEITSBRÜCKE VON USB ZU SATA

PONT À HAUT DÉBIT USB VERS SATA

(84) Designated Contracting States:
AL AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HR HU IE IS IT LI LT LU LV MC MK MT NL NO PL PT RO RS SE SI SK SM TR

(30) Priority: **19.10.2011 US 201113276371**
21.10.2010 US 405444 P

(43) Date of publication of application:
28.08.2013 Bulletin 2013/35

(73) Proprietor: **Marvell World Trade Ltd.**
St. Michael 14027 (BB)

(72) Inventors:
- **LIN, Chun-Lun**
  Santa Clara, CA 95134 (US)
- **TSAI, Kanting**
  San Jose, CA 95129 (US)
- **LAI, Dishi**
  San Jose, CA 95120 (US)
- **CHU, Hsi-Cheng**
  San Jose, CA 95131 (US)

(74) Representative: **Grünecker Patent- und Rechtsanwälte**
PartG mbB
Leopoldstraße 4
80802 München (DE)

(56) References cited:
US-A1- 2007 005 838
US-A1- 2010 185 808
US-B1- 7 620 747

EP 2 630 565 B1

Description

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application claims priority to U.S. Utility Application No. 13/276,371, filed on October 19, 2011, and the benefit of U.S. Provisional Application No. 61/405,444, filed on October 21, 2010.

FIELD

[0002] The present disclosure relates generally to data storage systems and more particularly to a Universal Serial Bus (USB) to Serial Advanced Technology Attachment (SATA) high-speed bridge for storage devices.

BACKGROUND

[0003] The background description provided herein is for the purpose of generally presenting the context of the disclosure. Work of the presently named inventors, to the extent the work is described in this background section, as well as aspects of the description that may not otherwise qualify as prior art at the time of filing, are neither expressly nor impliedly admitted as prior art against the present disclosure.

[0004] Referring now to FIG. 1, a data storage system 10 includes a storage device 12 and a host 14. The storage device 12 includes a disk controller 16 and a disk drive 18. The disk drive 18 may include a conventional disk drive that stores data on a magnetic medium, a solid-state disk that stores data on a semiconductor memory (e.g., a flash memory), or an optical disk drive that stores data on an optical disc.

[0005] The host 14 may include a laptop computer, a personal computer, or any other type of computing device. The storage device 12 may communicate with the host 14 via a Universal Serial Bus (USB) interface. Accordingly, the disk controller 16 may include a USB interface to communicate with the host 12. The disk controller 16 may also include a Serial Advanced Technology Attachment (SATA) interface to communicate with the disk drive 18.

[0006] US 2010/185808 A1 discloses methods and systems for storing and accessing data in UAS based flash memory device. UAS based flash memory device comprises a controller and a plurality of non-volatile memories (e.g., flash memory) it controls. Controller is configured for connecting to a UAS host via a physical layer (e.g., plug and wire based on USB 3.0) and for conducting data transfer operations via two sets of logical pipes. Controller further comprises a random-access-memory (RAM) buffer configured for enabling parallel and duplex data transfer operations through the sets of logical pipes. In addition, a Smart Storage Switch configured for connecting multiple non-volatile memory devices is included in the controller. Finally, a security module/engine/unit is provided for data security via user authentication data encryption/decryption of the device. Furthermore, the flash memory device includes an optical transceiver configured for optical connection to a host also configured with an optical transceiver.

[0007] US 2007/005838 A1 relates to serial ATA port addressing. This document discloses, in one aspect, a shared transport layer frame information structure (FIS) generation logic that may generate FISes for each of a plurality of SATA ports. This document discloses, in a further aspect, a port addressing logic, in communication with the shared transport layer FIS generation logic, that may select one of the SATA ports for each of the FISes.

[0008] US 7 620 747 B1 relates to software based native command queuing. This document discloses systems and methods for performing native command queuing according to the protocol specified by serial ATA II for transferring data between a disk and system memory. Native command queuing context for queued commands is maintained by a host controller device driver and is provided to the host controller as needed to process the queued commands. The host controller is simplified since it only stores the context of the one command being processed. The host controller generates a backoff interrupt when a command cannot be queued. The host controller generates a DMA transfer context request interrupt to request programming of the registers that store the context for the one command being processed.

[0009] It is the object of the present invention to provide a more efficient communication system and method between a host and a storage device.

[0010] This object is solved by the subject matter of the independent claims.

[0011] Preferred embodiments are defined by the dependent claims.

SUMMARY

[0012] A system includes a first controller configured to communicate with a host via a first interface; a second controller configured to communicate with a storage device via a second interface, where the second interface is different than the first interface; and a bridge module configured to allow the second controller to transfer data between the storage device and the host and to allow the second controller to access memory of the host via the first interface during the transfer.

[0013] In another feature, the bridge module is configured to provide a path for the transfer between the storage device and the host, where the path includes the first controller, the bridge module, and the second controller, and where the path excludes a buffer to store the data.

[0014] In another feature, the bridge module is configured to allow the second controller to convert an incoming command from the host while the transfer is in progress.

[0015] In another feature, the bridge module is configured return to the host a status of the transfer by interpreting information received from the second controller about the transfer.

[0016] In another feature, the first interface is a Universal Serial Bus (USB) interface, and the second interface is a Serial Advanced Technology Attachment (SATA) interface.

[0017] In another feature, the bridge module is configured to convert a first command received from the host to a second command, where the first command is compliant with the first interface, and where the second command is compliant with the second interface; and to transfer the second command to the second interface.

[0018] In another feature, the bridge module includes a bridge interface module configured to receive a request from the second controller for a data path providing access to the memory of the host. The bridge module includes an arbitration module configured to provide, based on the request, the data path to the second controller to transfer the data between the storage device and the memory of the host. The data path excludes a buffer to store the data. The bridge interface module allows the second controller to transfer the data between the storage device and the memory of the host via the data path.

[0019] In still other features, a system-on-chip (SOC) includes a Universal Serial Bus (USB) controller configured to interface a storage controller of a storage device to a host via a USB interface, and a Serial Advanced Technology Attachment (SATA) controller configured to interface the storage controller to the storage device via a SATA interface. The SOC further includes a command handling module configured to convert a USB command received by the USB controller from the host to a SATA command, and to output the SATA command to the SATA controller. The SATA controller is configured to generate a request for a data path for access to a memory of the host to transfer data between the storage device and the memory of the host according to the SATA command. The data path is a path between the memory and the storage device. The SOC further includes a data handling module configured to receive the request from the SATA controller, and to provide the data path to the SATA controller to transfer the data between the storage device and the memory of the host. The SATA controller is configured to transfer the data between the storage device and the memory of the host according to the SATA command.

[0020] In another feature, the SOC further includes a status handling module configured to receive and interpret command completion information from the SATA controller; to generate, based on the command completion, a status of the transfer by determining whether the transfer is completed without error; and to send the status of the transfer to the host via the USB controller.

[0021] In still other features, a method includes communicating with a host via a first interface of a first controller; communicating with a storage device via a second interface of a second controller, where the second interface is different than the first interface; allowing the second controller to transfer data between the storage device and the host; and allowing the second controller to access memory of the host via the first interface during the transfer.

[0022] In another feature, the method further includes providing a path for the transfer between the storage device and the host, where the path includes the first controller and the second controller, and where the path excludes a buffer to store the data.

[0023] In another feature, the method further includes converting an incoming command from the host while the transfer is in progress.

[0024] In another feature, the method further includes returning to the host a status of the transfer by interpreting information received from the second controller about the transfer.

[0025] In another feature, in the method, the first interface is a Universal Serial Bus (USB) interface, and the second interface is a Serial Advanced Technology Attachment (SATA) interface.

[0026] In another feature, the method further includes converting a first command received from the host to a second command, where the first command is compliant with the first interface, and where the second command is compliant with the second interface; and transferring the second command to the second interface.

[0027] In another feature, the method further includes receiving a request from the second controller for a data path providing access to the memory of the host; providing, based on the request, the data path to the second controller to transfer the data between the storage device and the memory of the host, where the data path excludes a buffer to store the data; and allowing the second controller to transfer the data between the storage device and the memory of the host via the data path.

[0028] Further areas of applicability of the present disclosure will become apparent from the detailed description, the claims and the drawings. The detailed description and specific examples are intended for purposes of illustration only and are not intended to limit the scope of the disclosure.

BRIEF DESCRIPTION OF DRAWINGS

[0029] The present disclosure will become more fully understood from the detailed description and the accompanying drawings, wherein:

FIG. 1 is a functional block diagram of a data storage system according to the prior art;

FIG. 2 is a functional block diagram of a disk controller that uses a buffer when transferring data between a host and a disk drive during read/write operations;

FIG. 3 is a functional block diagram of a disk controller that transfers data directly between a host and a disk drive during read/write operations without using a buffer;

FIG. 4 depicts data flow between a host and a disk

drive when the host issues a Universal Serial Bus (USB) Bulk-only Transport (BOT) command;

FIG. 5 depicts data flow between a host and a disk drive when the host issues a USB Attached SCSI (UAS) command; and

FIGs. 6A and 6B are flowcharts of methods for High-Speed bridge operations from command handling, data handling, status handling, and error handling perspectives.

DESCRIPTION

[0030] Referring now to FIG. 2, an example of a disk controller is shown. A disk controller 100 includes a processing module 102, a bus interface module 107, a buffer 104, a Universal Serial Bus (USB) controller 106, and a Serial Advanced Technology Attachment (SATA) controller 108. The USB controller 106 includes a USB interface module 110 that interfaces with the host 14 and a USB direct memory access (DMA) module 112 that communicates with the USB interface module 110, the processing module 102, and the buffer 104. The SATA controller 108 includes a SATA interface module 114 that interfaces with the disk drive 18 and a SATA DMA module 116 that communicates with the SATA interface module 114, the processing module 102, and the buffer 104. The bus interface module 107 communicates with the USB DMA module 112, the SATA DMA module 116, the processing module 102, and the buffer 104. The bus interface module 107 provides the USB DMA module 112 and the SATA DMA module 116 a DMA path to the buffer 104.

[0031] When the host 14 issues a write command to write data on the disk drive 18, the USB DMA module 112 receives the data from the host 14 via the USB interface module 110 and stores the data in the buffer 104. The SATA DMA module 116 retrieves the data from the buffer 104 and forwards the data to the SATA interface module 114, which writes the data to the disk drive 18.

[0032] When the host 14 issues a read command to read data from the disk drive 18, the SATA DMA module 116 receives the data from the disk drive 18 via the SATA interface module 114 and stores the data in the buffer 104. The USB DMA module 112 retrieves the data from the buffer 104 and forwards the data to the USB interface module 110, which outputs the data to the host 14.

[0033] Buffering data during read/write operations can decrease the rate at which data is transferred between the USB interface module 110 and the SATA interface module 114. Consequently, performance of high-speed interfaces such as USB 3.0 and SATA 3.0 can degrade despite having data transfer rates of 5 Gigabits per second (Gbps) and 6 Gbps, respectively. Moreover, buffering data during read/write operations generates many interrupts which the processing module 102 has to process and which can increase overhead and downgrade IO performance.

[0034] Buffering data during read/write operations can be eliminated by using a high-speed bridge instead of using a buffer between the USB and SATA controllers. As explained below, the bridge may work, for example, with Bulk-only Transport (BOT) and USB Attached SCSI (UAS) protocols of the USB interface and Native Command Queuing (NCQ) protocol of the SATA interface. The bridge determines a command phase, a data phase, and a status phase of data transfer between the USB and SATA controllers based on the BOT and UAS protocols. The command phase includes command conversion. The data phase includes data transfer according to the command. The status phase includes returning status of the command and associated data transfer to the host. The bridge can automatically convert an incoming command from the host. The SATA DMA module can directly receive/transmit data from/to the bridge without using the buffer. After determining that a data transaction is completed, the bridge returns status automatically based on the information received from the SATA controller. The bridge thus improves throughput and optimizes IO performance.

[0035] The present disclosure uses USB and SATA interfaces for example only. The teachings of the present disclosure are not limited to USB and SATA interfaces. The teachings of the present disclosure can be applied to other interfaces used to transfer data at high speed between a host and a disk drive.

[0036] Referring now to FIG. 3, a disk controller 200 includes a processing module 202, a USB controller 204, a bridge module 206, a USB DMA module 222, a bus interface module 208, and a SATA controller 108. The disk controller 200 can be implemented as an integrated circuit (IC) or a system-on-chip (SOC).

[0037] As explained below, a command handling module 220 in the bridge module 206 interrupts the processing module 202 when a USB command is received from the host 14. The processing module 202 converts the USB command to a SATA command. When the command is a write command to write data from the host 14 to the disk drive 18, the SATA controller 108 accesses the memory of the host 14 and transfers data from the memory of the host 14 directly to the disk drive 18. When the command is a read command to read data from the disk drive 18, the SATA controller 108 reads data from the disk drive and transfers data from the disk drive 18 directly to the memory of the host 14. The bridge module 206 allows the SATA controller 108 to directly receive/transmit data from/to memory of the host 14 without using the buffer 104 (not shown). The USB controller 204 includes the USB interface module 110, a receive FIFO 212, and a transmit FIFO 214. The USB interface module 110 stores the command received from the host 14 in the receive FIFO 212.

[0038] The bridge module 206 includes an arbitration module 218, a command handling module 220, a data handling module 224, a status handling module 228, and an interrupt handling module 230. The bridge module 206 does not include the buffer 104. The command han-

dling module 220 interrupts the processing module 202 when an incoming command is received from the host 14. The processing module 202 converts the command received from the host 14 to a SATA command and outputs the SATA command to the SATA controller 108 via the bus interface module 208. The SATA controller 108 is configured to transfer data according to the SATA command. After the SATA controller 108 is configured to transfer data according to the SATA command, the SATA controller 108 sends a request to the data handling module 224 for bus ownership to transfer data directly between the SATA controller 108 and the USB controller 204.

[0039] Depending on the type of command (e.g., read or write command), the arbitration module 218 arbitrates bus ownership and data flow between the USB DMA module 222 and the bridge interface module 206. The arbitration module 218 can also perform the arbitration for the command phase, data phase, and status phase automatically without software involvement.

[0040] During the read operation, the arbitration module 218 arbitrates bus ownership and data flow from the data handling module 224 to the transmit FIFO 214 to provide a direct data path for the SATA DMA module 116 to transfer the data read from the disk drive 18 directly to the memory of the host 14 via the USB controller 204.

[0041] During the write operation, the arbitration module 218 arbitrates bus ownership and data flow from the receive FIFO 212 and the data handling module 224 to provide a direct data path for the SATA DMA module 116 to transfer the data from the memory of the host 14 directly to the disk drive 18 via the USB controller 204.

[0042] When each read/write command is completed, the SATA interface module 114 provides information directly to the status handling module 228. The status handling module 228 can interpret the information and send out status automatically without software intervention once the data phase is completed without error.

[0043] While only one SATA controller is shown for simplicity of illustration, a plurality of SATA controllers can be interfaced with the bus interface module 208. For each additional SATA controller interfaced with the bus interface module 208, the bridge module 206 can include an additional set of the data handling module 224 and the status handling module 228. The USB DMA module 222 handles DMA for the plurality of SATA controllers and allows each SATA controller to transfer data using a corresponding set of the data handling module 224 and the status handling module 228.

[0044] The data handling module 224 provides a direct data path for the SATA DMA module 116 to receive/transmit data directly from/to the memory of the host 14 without storing the data in the buffer 104. The data handling module 224 actively transfers data in and out of the USB controller 204 according to the SATA command. The status handling module 228 also generates status information for a command when the status handling module 228 receives command completion information directly from the module SATA interface module 114 after the data transfer corresponding to the command is completed. If an error occurs at the USB controller 204, the interrupt handling module 230 notifies the processing module 202.

[0045] Referring now to FIG. 4, data flows 300 for a USB Bulk-only Transport (BOT) command are shown. At 302, the host 14 issues a USB BOT command. At 304, the USB controller 204 receives the USB BOT command. At 306, the USB DMA module 222 or the command handling module 220 interrupts the processing module 202. The processing module 202 converts the USB BOT command to a SATA command and sends the SATA command to the SATA controller 108. At 308, the SATA controller 108 receives the SATA command. The SATA controller 108 prepares to transfer data according to the SATA command.

[0046] At 310, the SATA controller 108 transfers data directly from the host 14 to the disk drive 18 if the USB command is a write command. At 312, the SATA controller 108 transfers data directly from the disk drive 18 to the host 14 if the USB command is a read command.

[0047] At 314, the SATA controller 108 interrupts the processing module 202 when the data transfer is completed. At 318, at the same time, the SATA interface module 114 sends information directly to the status handling module 228. The status handling module 228 interprets the information and generates status without involvement of the processing module 202. At 320, the host 14 receives the status information.

[0048] Referring now to FIG. 5, data flows 400 for USB Attached SCSI (UAS) commands are shown. At 402, the host 14 issues USB UAS commands (e.g., a burst of up to 32 commands at a time per one SATA port). At 404, the USB controller 204 receives the USB UAS commands. The USB UAS commands are not merged together. At 406, the USB DMA module 222 or the command handling module 220 interrupts the processing module 202 (e.g., up to 32 times).

[0049] The processing module 202 converts the USB UAS commands to SATA commands (one-to-one conversion) and sends the SATA commands to the SATA controller 108 (e.g., up to 32 at a time). At 408, the SATA controller 108 receives the SATA commands (e.g., up to 32 at a time). The SATA controller 108 prepares to transfer data according to the SATA commands. The SATA controller 108 sets up one DMA operation for the data transfers (read or write operations) of all the SATA commands.

[0050] At 410, the SATA controller 108 transfers data directly from the host 14 to the disk drive 18 if the USB commands are write commands. For example, the SATA controller 108 performs up to 32 write operations (data transfers). At 412, the SATA controller 108 transfers data directly from the disk drive 18 to the host 14 if the USB commands are read commands. For example, the SATA controller 108 performs up to 32 read operations (data transfers).

[0051] At 414, the SATA controller 108 interrupts the

processing module 202 when the data transfers for the SATA commands are completed (e.g., up to 32 at a time). At 416, the processing module 202 receives the interrupts (e.g., up to 32 at a time). At 418, at the same time, the SATA interface module 114 sends information directly to the status handling module 228 (e.g., up to 32 at a time). The status handling module 228 interprets the information and generates status without involvement of the processing module 202. At 420, the host 14 receives the status information (e.g., up to 32 status information at a time).

[0052] Referring now to FIG. 6A, a flowchart for command handling according to the present disclosure is shown. At 502, the command handling module 220 is idle after power up. At 504, the command handling module 220 receives a USB read/write command from host 14. At 508, the command handling module 220 notifies the processing module 202, and the processing module 202 converts the received USB command to a SATA command. At 510, the processing module 202 initiates a SATA read/write operation. At 512, data is transferred directly between the high-speed bridge and the SATA controller 108. During the direct data transfer between the high-speed bridge and the SATA controller 108, if the command handling module 220 receives a next command, the command handling module 220 notifies the processing module 202, which processes the next command in parallel to the previous command for which the data is being transferred.

[0053] At 514, a determination is made if an error occurred during data transfer and/or command processing (e.g., at 504, 508, 510, and/or 512). At 516, if an error occurred during data transfer and/or command processing (e.g., at 504, 508, 510, and/or 512), the interrupt handling module 230 takes charge and notifies the processing module 202 for further instruction, and the command handling module 220 returns to the idle state.

[0054] Referring now to FIG. 6B, a method 600 for status handling according to the present disclosure is shown. At 602, the status handling module 228 is idle after power on. At 604, when the SATA controller 108 completes any read/write command, the SATA controller 108 interrupts the processing module 202. At the same time, the SATA interface module 114 notifies and sends command completion information directly to the status handling module 228. At 606, a determination is made if data transfer for current command is completed. At 608, if data transfer for current command is completed, a determination is made if the data transaction is completed without error. At 610, the status handling module 228 sends out status without involvement of the processing module 202. At 612, if an error occurred in the data transaction, the interrupt handling module 230 notifies the processing module 202 for further instruction.

[0055] For purposes of clarity, the same reference numbers will be used in the drawings to identify similar elements. As used herein, the phrase at least one of A, B, and C should be construed to mean a logical (A or B or C), using a non-exclusive logical OR. It should be understood that one or more steps within a method may be executed in different order (or concurrently) without altering the principles of the present disclosure.

[0056] As used herein, the term module may refer to, be part of, or include an Application Specific Integrated Circuit (ASIC); an electronic circuit; a combinational logic circuit; a field programmable gate array (FPGA); a processor (shared, dedicated, or group) that executes code; other suitable hardware components that provide the described functionality; or a combination of some or all of the above, such as in a system-on-chip. The term module may include memory (shared, dedicated, or group) that stores code executed by the processor.

[0057] The term code, as used above, may include software, firmware, and/or microcode, and may refer to programs, routines, functions, classes, and/or objects. The term shared, as used above, means that some or all code from multiple modules may be executed using a single (shared) processor. In addition, some or all code from multiple modules may be stored by a single (shared) memory. The term group, as used above, means that some or all code from a single module may be executed using a group of processors. In addition, some or all code from a single module may be stored using a group of memories.

[0058] The apparatuses and methods described herein may be implemented by one or more computer programs executed by one or more processors. The computer programs include processor-executable instructions that are stored on a non-transitory tangible computer readable medium. The computer programs may also include stored data. Non-limiting examples of the non-transitory tangible computer readable medium are non-volatile memory, magnetic storage, and optical storage.

[0059] The following is a list of further preferred embodiments, which are however not part of the invention:

Embodiment 1: A system comprising:

a first controller configured to communicate with a host via a first interface;

a second controller configured to communicate with a storage device via a second interface, wherein the second interface is different than the first interface; and

a bridge module configured to allow the second controller to transfer data between the storage device and the host and to allow the second controller to access memory of the host via the first interface during the transfer.

Embodiment 2: The system of embodiment 1, wherein the bridge module is configured to provide a path for the transfer between the storage device and the host, wherein the path includes the first controller,

the bridge module, and the second controller, and wherein the path excludes a buffer to store the data.

Embodiment 3: The system of embodiment 1, wherein the bridge module is configured to allow the second controller to convert an incoming command from the host while the transfer is in progress.

Embodiment 4: The system of embodiment 1, wherein the bridge module is configured return to the host a status of the transfer by interpreting information received from the second controller about the transfer.

Embodiment 5: The system of embodiment 1, wherein:

the first interface is a Universal Serial Bus (USB) interface; and

the second interface is a Serial Advanced Technology Attachment (SATA) interface.

Embodiment 6: The system of embodiment 1, wherein the bridge module is configured to:

convert a first command received from the host to a second command, wherein the first command is compliant with the first interface, and wherein the second command is compliant with the second interface; and

transfer the second command to the second interface.

Embodiment 7: The system of embodiment 6, further comprising the second interface configured to:

request the bridge module for a data path providing access to the memory of the host, wherein the data path excludes a buffer to store the data;

transfer the data from the memory of the host to the storage device via the data path according to the second command in response to the first command being a write command to write the data to the storage device; and

transfer the data from the storage device to the memory of the host according to the second command in response to the first command being a read command to read the data from the storage device.

Embodiment 8: The system of embodiment 1, wherein the bridge module comprises:

a bridge interface module configured to receive a request from the second controller for a data path providing access to the memory of the host; and

an arbitration module configured to provide, based on the request, the data path to the second controller to transfer the data between the storage device and the memory of the host,

wherein the data path excludes a buffer to store the data, and

wherein the bridge interface module allows the second controller to transfer the data between the storage device and the memory of the host via the data path.

Embodiment 9: A system-on-chip (SOC) comprising:

a Universal Serial Bus (USB) controller configured to interface a storage controller of a storage device to a host via a USB interface;

a Serial Advanced Technology Attachment (SATA) controller configured to interface the storage controller to the storage device via a SATA interface;

a command handling module configured to

convert a USB command received by the USB controller from the host to a SATA command, and

output the SATA command to the SATA controller,

wherein the SATA controller is configured to generate a request for a data path for access to a memory of the host to transfer data between the storage device and the memory of the host according to the SATA command, and

wherein the data path is a path between the memory and the storage device; and

a data handling module configured to

receive the request from the SATA controller, and

provide the data path to the SATA controller to transfer the data between the storage device and the memory of the host,

wherein the SATA controller is configured to transfer the data between the storage device

and the memory of the host according to the SATA command.

Embodiment 10: The SOC of embodiment 9, further comprising a status handling module configured to:

receive and interpret command completion information from the SATA controller, and

generate, based on the command completion, a status of the transfer by determining whether the transfer is completed without error, and

send the status of the transfer to the host via the USB controller.

Embodiment 11: A method comprising:

communicating with a host via a first interface of a first controller;

communicating with a storage device via a second interface of a second controller, wherein the second interface is different than the first interface;

allowing the second controller to transfer data between the storage device and the host; and

allowing the second controller to access memory of the host via the first interface during the transfer.

Embodiment 12: The method of embodiment 11, further comprising providing a path for the transfer between the storage device and the host, wherein the path includes the first controller and the second controller, and wherein the path excludes a buffer to store the data.

Embodiment 13: The method of embodiment 11, further comprising converting an incoming command from the host while the transfer is in progress.

Embodiment 14: The method of embodiment 11, further comprising returning to the host a status of the transfer by interpreting information received from the second controller about the transfer.

Embodiment 15: The method of embodiment 11, wherein:

the first interface is a Universal Serial Bus (USB) interface; and

the second interface is a Serial Advanced Technology Attachment (SATA) interface.

Embodiment 16: The method of embodiment 11, further comprising:

converting a first command received from the host to a second command, wherein the first command is compliant with the first interface, and wherein the second command is compliant with the second interface; and

transferring the second command to the second interface.

Embodiment 17: The method of embodiment 16, further comprising:

requesting a data path providing the second controller access to the memory of the host, wherein the data path excludes a buffer to store the data;

transferring the data from the memory of the host to the storage device via the data path according to the second command in response to the first command being a write command to write the data to the storage device; and

transferring the data from the storage device to the memory of the host according to the second command in response to the first command being a read command to read the data from the storage device.

Embodiment 18: The method of embodiment 11, further comprising:

receiving a request from the second controller for a data path providing access to the memory of the host;

providing, based on the request, the data path to the second controller to transfer the data between the storage device and the memory of the host, wherein the data path excludes a buffer to store the data; and

allowing the second controller to transfer the data between the storage device and the memory of the host via the data path.

Claims

1. A system (200) comprising:

a first controller (204) configured to communicate with a host (14) via a first interface (110);
a second controller (108) configured to communicate with a storage device (18) via a second

interface (114), wherein the second interface is of a different type than the first interface; and
a bridge module (206) configured to

allow the second controller to transfer data between the storage device and the host and to allow the second controller to access memory of the host via the first interface during the transfer, and
provide a data path for the transfer between the storage device and the host, wherein the data path includes the first controller, the bridge module, and the second controller, and wherein the data path between the first and second controllers excludes a buffer to store the data,

wherein the second controller is further configured to use the provided data path to receive/transmit data directly from/to the memory of the host to/from the storage device without storing the data in a buffer between the first and second controllers; and
wherein the bridge module (206) comprises:

a command handling module (220) configured to, during a command phase of data transfer between the first interface (110) and the second interface (114):

convert a first command received by the first interface (110) from the host (14) to a second command, and
output the second command to the second interface (114),
wherein the second interface (114) is configured to generate a request for a data path for access to a memory of the host (14) to transfer data between the storage device (18) and the memory of the host (14) according to the second command;

a data handling module (224) configured to, during a phase of the data transfer between the first interface (110) and the second interface (114):

receive the request from the second interface (114), and
provide the data path to the second interface (114) to transfer the data between the storage device (18) and the memory of the host (14) according to the second command without buffering the data in the bridge module (206);

a status handling module (228) configured to, during a status phase of the data transfer between the first interface (110) and the second interface (114):

receive and interpret command completion from the second interface (114),
generate, based on the command completion information, a status of the data transfer by determining whether the data transfer is completed without error, and
send the status of the data transfer to the host via the first interface (110); and

an arbitration module (218) configured to:

arbitrate bus ownership between the first interface (110) and the second interface (114) depending on whether the first command is a read command or a write command, and
arbitrate bus ownership between the first interface (110) and the second interface (114) for the command phase, the data phase, and the status phase of the data transfer according to the second command.

2. The system (200) of claim 1, wherein the bridge module (206) is configured to allow the second controller (108) to convert the first command received from the host (14) while the data transfer is in progress.

3. The system (200) of claim 1, wherein:

the first interface (110) is a Universal Serial Bus, USB, interface; and
the second interface (114) is a Serial Advanced Technology Attachment, SATA, interface.

4. The system (200) of claim 1, wherein the first command is compliant with the first interface (110), and wherein the second command is compliant with the second interface (114).

5. The system (200) of claim 1, wherein the second interface (114) is configured to:

transfer the data from the memory of the host to the storage device (18) via the data path according to the second command in response to the first command being a write command to write the data to the storage device; and
transfer the data from the storage device to the memory of the host according to the second command in response to the first command being a read command to read the data from the storage device.

6. The system (200) of claim 1, wherein the system is a system-on-chip, SOC.

7. A method comprising:

communicating with a host (14) via a first interface (110) of a first controller (204);
communicating with a storage device (18) via a second interface (114) of a second controller (108), wherein the second interface is of a different type than the first interface;
allowing the second controller to transfer data between the storage device and the host;
allowing the second controller to access memory of the host via the first interface during the transfer; and
providing a data path for the transfer between the storage device and the host, wherein the data path includes the first controller and the second controller, wherein the data path between the first and second controllers excludes a buffer to store the data,
allowing the second controller to use the provided data path to receive/transmit data directly from/to the memory of the host to/from the storage device without storing the data in a buffer between the first and second controllers;
communicating with the first interface (110) and the second interface (114) via a bridge module (206);
during a command phase of data transfer between the first interface (110) and the second interface (114):

converting a first command received by the first interface (110) from the host to a second command,
outputting the second command to the second interface (114), and
generating, from the second interface (114) a request for a data path to a memory of the host(14) to transfer data between the storage device (18) and the memory of the host (14) according to the second command;

during a data phase of the data transfer between the first interface (110) and the second interface (114):

receiving the request from the second interface (114), and
providing the data path to the second interface (114) to transfer the data between the storage device (18) and the memory of the host (14) according to the second command without buffering the data in the bridge module (206);

during a status phase of the data transfer between the first interface (110) and the second interface (114):

receiving and interpreting command completion information from the second interface (114),
generating, based on the command completion information, a status of data transfer by determining whether the data transfer is completed without error, and
sending the status of the data transfer to the host (14) via the first interface;

arbitrating bus ownership between the first interface (110) and the second interface (114) depending on whether the first command is a read command or a write command; and
arbitrating bus ownership between the first interface (110) and the second interface (114) for the command phase, the data phase, and the status phase of the data transfer according to the second command.

8. The method of claim 7, further comprising converting the first command from the host (14) while the data transfer is in progress.

9. The method of claim 7, wherein:

the first interface (110) is a Universal Serial Bus, USB, interface; and
the second interface (114) is a Serial Advanced Technology Attachment, SATA, interface.

10. The method of claim 7, wherein the first command is compliant with the first interface (110), and the second command is compliant with the second interface (114); and
wherein the method further comprises:

transferring the data from the memory of the host to the storage device (18) via the data path according to the second command in response to the first command being a write command to write the data to the storage device; and
transferring the data from the storage device to the memory of the host according to the second command in response to the first command being a read command to read the data from the storage device.

Patentansprüche

1. System (200), mit:

einer ersten Steuerung (204), die ausgebildet ist, mit einem Host (14) über eine erste Schnittstelle (110) zu kommunizieren;
einer zweiten Steuerung (108), die ausgebildet ist, mit einer Speichereinrichtung (18) über eine

zweite Schnittstelle (114) zu kommunizieren, wobei die zweite Schnittstelle von einer anderen Art als die erste Schnittstelle ist; und
einem Brückenmodul (206), das ausgebildet ist zum

Ermöglichen für die zweite Steuerung, Daten zwischen der Speichereinrichtung und dem Host auszutauschen, und Ermöglichen für die zweite Steuerung, auf einen Speicher des Hosts über die erste Schnittstelle während des Austausches zuzugreifen; und
Bereitstellen eines Datenkanals für den Austausch zwischen der Speichereinrichtung und dem Host, wobei der Datenkanal die erste Steuerung, das Brückenmodul und die zweite Steuerung umfasst, und wobei der Datenkanal zwischen der ersten und der zweiten Steuerung ohne einen Puffer zur Speicherung der Daten vorgesehen ist,

wobei die zweite Steuerung ferner ausgebildet ist, den bereitgestellten Datenkanal zum Empfangen/Senden von Daten direkt aus/zu dem Speicher des Hosts zu/aus der Speichereinrichtung ohne Speicherung der Daten in einem Puffer zwischen der ersten und der zweiten Steuerung zu verwenden; und
wobei das Brückenmodul (206) aufweist:

ein Befehlshandhabungsmodul (220), das ausgebildet ist, während einer Befehlsphase eines Datenaustausches zwischen der ersten Schnittstelle (110) und der zweiten Schnittstelle (114):

einen von der ersten Schnittstelle (110) aus dem Host (14) empfangenen ersten Befehl in einen zweiten Befehl umzuwandeln, und
den zweiten Befehl an die zweite Schnittstelle (114) auszugeben,
wobei die zweite Schnittstelle (114) ausgebildet ist, eine Anforderung bezüglich eines Datenkanals zum Zugriff auf einen Speicher des Hosts (14) zum Austausch von Daten zwischen der Speichereinrichtung (18) und dem Speicher des Hosts (14) gemäß dem zweiten Befehl zu erzeugen;

ein Datenhandhabungsmodul (224), das während einer Phase des Datenaustausches zwischen der ersten Schnittstelle (110) und der zweiten Schnittstelle (114) ausgebildet ist:

die Anforderung aus der zweiten Schnittstelle (114) zu empfangen, und den Datenkanal für die zweite Schnittstelle (114) zum Austausch der Daten zwischen der Speichereinrichtung (18) und dem Speicher des Hosts (14) entsprechend dem zweiten Befehl ohne Zwischenspeicherung der Daten in dem Brückenmodul (206) bereitzustellen;

ein Statushandhabungsmodul 8228), das während einer Statusphase des Datenaustausches zwischen der ersten Schnittstelle 8110) und der zweiten Schnittstelle (114) ausgebildet ist:

eine Befehlsvervollständigungsinformation aus der zweiten Schnittstelle (114) zu empfangen und auszuwerten, auf der Grundlage der Befehlsvervollständigungsinformation einen Status des Datenaustausches zu erzeugen, indem ermittelt wird, ob der Datenaustausch ohne Fehler abgeschlossen ist, und
den Status des Datenaustausches über die erste Schnittstelle (110) an den Host zu senden; und

ein Verteilungsmodul (218), das ausgebildet ist:

einen Busbesitz zwischen der ersten Schnittstelle (110) und der zweiten Schnittstelle (114) abhängig davon aufzuteilen, ob der erste Befehl ein Lesebefehl oder ein Schreibbefehl ist, und den Busbesitz zwischen der ersten Schnittstelle (110) und der zweiten Schnittstelle (114) für die Befehlsphase, die Datenphase und die Statusphase des Datenaustausches entsprechend dem zweiten Befehl aufzuteilen.

2. System (200) nach Anspruch 1, wobei das Brückenmodul (206) ausgebildet ist, es der zweiten Steuerung (108) zu ermöglichen, den aus dem Host (14) empfangenen ersten Befehl umzuwandeln, während der Datenaustausch im Gange ist.

3. System (200) nach Anspruch 1, wobei:

die erste Schnittstelle (110) eine Schnittstelle für den universellen seriellen Bus, USB, ist; und
die zweite Schnittstelle (114) eine Schnittstelle für den seriellen erweiterten Technikanhang, SATA, ist.

4. System (200) nach Anspruch 1, wobei der erste Befehl zu der ersten Schnittstelle (110) konform ist, und wobei der zweite Befehl zu der zweiten Schnittstelle (114) konform ist.

5. System (200) nach Anspruch 1, wobei die zweite Schnittstelle (114) ausgebildet ist zum:

Übertragen der Daten aus dem Speicher des Hosts zu der Speichereinrichtung (18) über den Datenkanal gemäß dem zweiten Befehl in Reaktion darauf, dass der erste Befehl ein Schreibbefehl zum Schreiben der Daten in die Speichereinrichtung ist; und
Übertragen der Daten aus der Speichereinrichtung zu dem Speicher des Hosts gemäß dem zweiten Befehl in Reaktion darauf, dass der erste Befehl ein Lesebefehl zum Auslesen der Daten aus der Speichereinrichtung ist.

6. System (200) nach Anspruch 1, wobei das System ein System-auf-Chip, SOC, ist.

7. Verfahren, mit:

Kommunizieren mit einem Host 814) über eine erste Schnittstelle (110) einer ersten Steuerung (204);
Kommunizieren mit einer Speichereinrichtung (18) über eine zweite Schnittstelle (114) einer zweiten Steuerung (108), wobei die zweite Schnittstelle von anderer Art als die erste Schnittstelle ist;
Ermöglichen, dass die zweite Steuerung einen Datenaustausch zwischen der Speichereinrichtung und dem Host ausführt;
Ermöglichen, dass die zweite Steuerung während des Austausches auf einen Speicher des Hosts über die erste Schnittstelle zugreift; und
Bereitstellen eines Datenkanals für den Austausch zwischen der Speichereinrichtung und dem Host, wobei der Datenkanal die erste Steuerung und die zweite Steuerung enthält, und wobei der Datenkanal zwischen der ersten und der zweiten Steuerung ohne einen Puffer zur Speicherung der Daten vorgesehen ist,
Ermöglichen, dass die zweite Steuerung den bereitgestellten Datenkanal verwendet, um Daten aus/von dem Speicher des Hosts zu/aus der Speichereinrichtung direkt zu empfangen/zu senden ohne Zwischenspeicherung der Daten in einem Puffer zwischen der ersten und der zweiten Steuerung;
Kommunizieren mit der ersten Schnittstelle (110) und der zweiten Schnittstelle (114) über ein Brückenmodul (206);
während einer Befehlsphase des Datenaustausches zwischen der ersten Schnittstelle (110) und der zweiten Schnittstelle (114):

Umwandeln eines von der ersten Schnittstelle (110) aus dem Host empfangenen ersten Befehls in einen zweiten Befehl,
Ausgeben des zweiten Befehls an die zweite Schnittstelle (114), und
Erzeugen, aus der zweiten Schnittstelle (114), einer Anforderung bezüglich eines Datenkanals zu einem Speicher des Hosts (14), um Daten zwischen der Speichereinrichtung (18) und dem Speicher des Hosts (14) auszutauschen, gemäß dem zweiten Befehl;

während einer Datenphase des Datenaustausches zwischen der ersten Schnittstelle (110) und der zweiten Schnittstelle (114):

Empfangen der Anforderung aus der zweiten Schnittstelle (114), und
Bereitstellen des Datenkanals für die zweite Schnittstelle (114) zum Austausch der Daten zwischen der Speichereinrichtung (18) und dem Speicher des Hosts (14) gemäß dem zweiten Befehl ohne Zwischenspeicherung der Daten in dem Brückenmodul (206);

während einer Statusphase des Datenaustausches zwischen der ersten Schnittstelle (110) und der zweiten Schnittstelle (114):

Empfangen und Auswerten einer Befehlsvervollständigungsinformation aus der zweiten Schnittstelle (114),
Erzeugen, auf der Grundlage der Befehlsvervollständigungsinformation, eines Status eines Datenaustausches durch Ermitteln, ob der Datenaustausch ohne Fehler abgeschlossen ist, und
Senden des Status des Datenaustausches über die erste Schnittstelle an den Host (14);

Aufteilen eines Busbesitzes auf die erste Schnittstelle (110) und die zweite Schnittstelle (114) in Abhängigkeit davon, ob der erste Befehl ein Lesebefehl oder ein Schreibbefehl ist; und
Aufteilen des Busbesitzes auf die erste Schnittstelle (110) und die zweite Schnittstelle (114) für die Befehlsphase, die Datenphase und die Statusphase des Datenaustausches entsprechend dem zweiten Befehl.

8. Verfahren nach Anspruch 7, das ferner umfasst: Umwandeln des ersten Befehls aus dem Host (14), während der Datenaustausch in Gange ist.

9. Verfahren nach Anspruch 7, wobei:

die erste Schnittstelle (110) eine Schnittstelle des universellen seriellen Busses, USB, ist; und die zweite Schnittstelle (114) eine Schnittstelle des seriellen erweiterten Technikanhangs, SATA, ist.

10. Verfahren nach Anspruch 7, wobei der erste Befehl zu der ersten Schnittstelle (110) konform ist und der zweite Befehl zu der zweiten Schnittstelle (114) konform ist; und
wobei das Verfahren ferner umfasst:

Übertragen der Daten aus dem Speicher des Hosts zu der Speichereinrichtung (18) über den Datenkanal gemäß dem zweiten Befehl in Reaktion darauf, dass der erste Befehl ein Schreibbefehl zum Schreiben der Daten in die Speichereinrichtung ist; und
Übertragen der Daten aus der Speichereinrichtung zu dem Speicher des Hosts gemäß dem zweiten Befehl in Reaktion darauf, dass der erste Befehl ein Lesebefehl zum Auslesen der Daten aus der Speichereinrichtung ist.

Revendications

1. Système (200) comprenant:

un premier contrôleur (204) configuré pour communiquer avec un hôte (14) via une première interface (110);
un second contrôleur (108) configuré pour communiquer avec un dispositif de stockage (18) via une seconde interface (114), dans lequel la seconde interface est d'un type différent de la première interface; et
un module de pont (206) configuré pour

permettre au second contrôleur de transférer des données entre le dispositif de stockage et l'hôte et permettre au second contrôleur d'accéder à la mémoire de l'hôte via la première interface pendant le transfert, et fournir un chemin de données pour le transfert entre le dispositif de stockage et l'hôte, dans lequel le chemin de données comprend le premier contrôleur, le module de pont et le second contrôleur, et dans lequel le chemin de données entre le premier contrôleur et le second contrôleur exclut un tampon pour stocker les données,

dans lequel le second contrôleur est en outre configuré pour utiliser le chemin de données fourni pour recevoir/transmettre des données directement depuis/ vers la mémoire de l'hôte vers/depuis le dispositif de stockage sans stocker les données dans un tampon entre le premier contrôleur et le second contrôleur; et
dans lequel le module de pont (206) comprend:

un module de gestion de commande (220) configuré pour, pendant une phase de commande de transfert de données entre la première interface (110) et la seconde interface (114):

convertir une première commande reçue par la première interface (110) de l'hôte (14) en une seconde commande, et
délivrer la seconde commande à la seconde interface (114),
dans lequel la seconde interface (114) est configurée pour générer une requête portant sur un chemin de données pour accéder à une mémoire de l'hôte (14) afin de transférer des données entre le dispositif de stockage (18) et la mémoire de l'hôte (14) selon la seconde commande;

un module de traitement de données (224) configuré pour, pendant une phase de transfert de données entre la première interface (110) et la seconde interface (114):

recevoir la requête provenant de la seconde interface (114), et
fournir le chemin de données à la seconde interface (114) pour transférer les données entre le dispositif de stockage (18) et la mémoire de l'hôte (14) selon la seconde commande sans mettre en mémoire tampon les données dans le module de pont (206);

un module de traitement d'état (228) configuré pour, pendant une phase d'état du transfert de données entre la première interface (110) et la seconde interface (114):

recevoir et interpréter l'achèvement de la commande à partir de la seconde interface (114),
générer, sur la base des informations d'achèvement de commande, un état du transfert de données en déterminant si le transfert de données est terminé sans erreur, et
envoyer l'état du transfert de données à l'hôte via la première interface (110); et

un module d'arbitrage (218) configuré pour:

arbitrer la propriété de bus entre la première interface (110) et la seconde interface (114) selon que la première commande est une commande de lecture ou une commande d'écriture, et
arbitrer la propriété de bus entre la première interface (110) et la seconde interface (114) pour la phase de commande, la phase de données et la phase d'état du transfert de données selon la seconde commande.

2. Système (200) selon la revendication 1, dans lequel le module de pont (206) est configuré pour permettre au second contrôleur (108) de convertir la première commande reçue de l'hôte (14) pendant que le transfert de données est en cours.

3. Système (200) selon la revendication 1, dans lequel:

la première interface (110) est une interface USB (Universal Serial Bus); et
la seconde interface (114) est une interface SATA, (Sériai Advanced Technology Attachment).

4. Système (200) selon la revendication 1, dans lequel la première commande est conforme à la première interface (110), et dans lequel la seconde commande est conforme à la seconde interface (114).

5. Système (200) selon la revendication 1, dans lequel la seconde interface (114) est configurée pour:

transférer les données de la mémoire de l'hôte vers le dispositif de stockage (18) via le chemin de données conformément à la seconde commande en réponse à la première commande qui est une commande d'écriture pour écrire les données dans le dispositif de stockage; et
transférer les données du dispositif de stockage vers la mémoire de l'hôte conformément à la seconde commande en réponse à la première commande qui est une commande de lecture pour lire les données provenant du dispositif de stockage.

6. Système (200) selon la revendication 1, dans lequel le système est un système sur puce, SOC.

7. Procédé consistant à:

communiquer avec un hôte (110) via une première interface (204) d'un premier contrôleur (204);
communiquer avec un dispositif de stockage (18) via une seconde interface (114) d'un second contrôleur (108), dans lequel la seconde interface est d'un type différent de la première interface;
permettre au second contrôleur de transférer des données entre le dispositif de stockage et l'hôte;
permettre au second contrôleur d'accéder à la mémoire de l'hôte via la première interface pendant le transfert; et
fournir un chemin de données pour le transfert entre le dispositif de stockage et l'hôte, dans lequel le chemin de données comprend le premier contrôleur et le second contrôleur, et dans lequel le chemin de données entre le premier contrôleur et le second contrôleur exclut un tampon pour stocker les données,
permettre au second contrôleur d'utiliser le chemin de données fourni pour recevoir/transmettre des données directement depuis/vers la mémoire de l'hôte vers/depuis le dispositif de stockage sans stocker les données dans un tampon entre le premier contrôleur et le second contrôleur;
communiquer avec la première interface (110) et la second interface (114) via un module de pont (206);
pendant une phase de commande de transfert de données entre la première interface (110) et la seconde interface (114):

convertir une première commande reçue par la première interface (110) en provenance de l'hôte en une seconde commande,
délivrer la second commande à la seconde interface (114), et
générer, à partir de la seconde interface (114), une requête portant sur un chemin de données vers une mémoire de l'hôte (14) afin de transférer des données entre le dispositif de stockage (18) et la mémoire de l'hôte (14) selon la seconde commande;

pendant une phase de transfert de données entre la première interface (110) et la seconde interface (114):

recevoir la requête provenant de la seconde interface (114), et
fournir le chemin de données à la seconde interface (114) pour transférer les données entre le dispositif de stockage (18) et la mémoire de l'hôte (14) selon la seconde commande sans mettre en mémoire tampon les données dans le module de pont (206);

pendant une phase d'état du transfert de données entre la première interface (110) et la seconde interface (114):

recevoir et interpréter l'achèvement de la commande à partir de la seconde interface (114),
générer, sur la base des informations d'achèvement de commande, un état du transfert de données en déterminant si le transfert de données est terminé sans erreur, et
envoyer l'état du transfert de données à l'hôte (14) via la première interface;

arbitrer la propriété de bus entre la première interface (110) et la seconde interface (114) selon que la première commande est une commande de lecture ou une commande d'écriture, et
arbitrer la propriété de bus entre la première interface (110) et la seconde interface (114) pour la phase de commande, la phase de données et la phase d'état du transfert de données selon la seconde commande.

8. Procédé selon la revendication 7, comprenant en outre la conversion de la première commande de l'hôte (14) pendant que le transfert de données est en cours

9. Procédé selon la revendication 7, dans lequel:

la première interface (110) est une interface USB (Universal Serial Bus); et
la seconde interface (114) est une interface SATA, (Sériai Advanced Technology Attachment).

10. Procédé selon la revendication 7, dans lequel la première commande est conforme à la première interface (110), et dans lequel la seconde commande est conforme à la seconde interface (114),
et dans lequel le procédé consiste en outre à:

transférer les données de la mémoire de l'hôte vers le dispositif de stockage (18) via le chemin de données conformément à la seconde commande en réponse à la première commande qui est une commande d'écriture pour écrire les données dans le dispositif de stockage; et
transférer les données du dispositif de stockage vers la mémoire de l'hôte conformément à la seconde commande en réponse à la première commande qui est une commande de lecture pour lire les données provenant du dispositif de stockage.

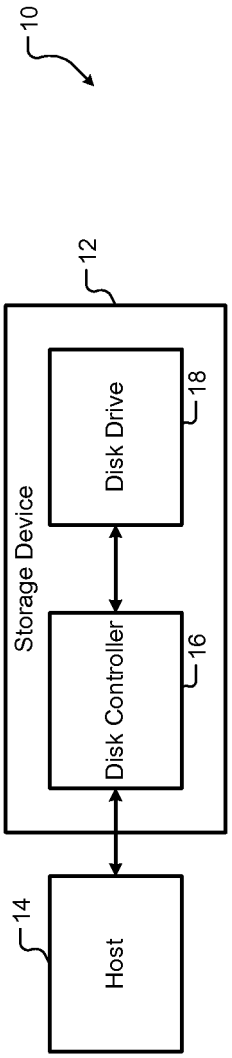

FIG. 1
Prior Art

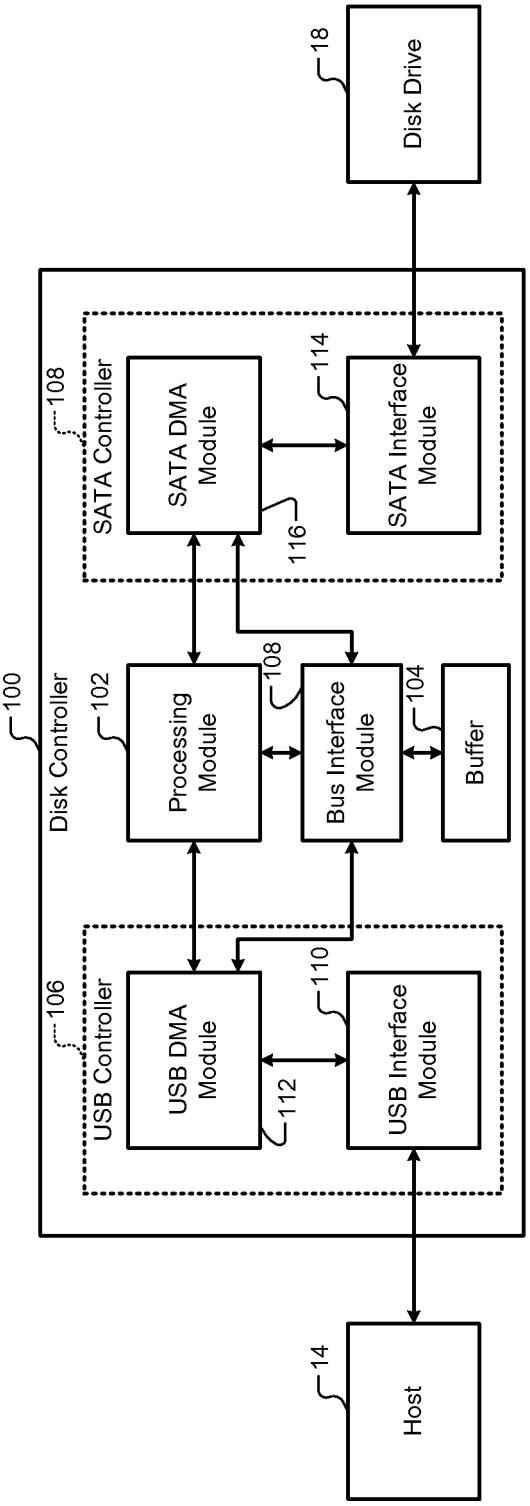

FIG. 2

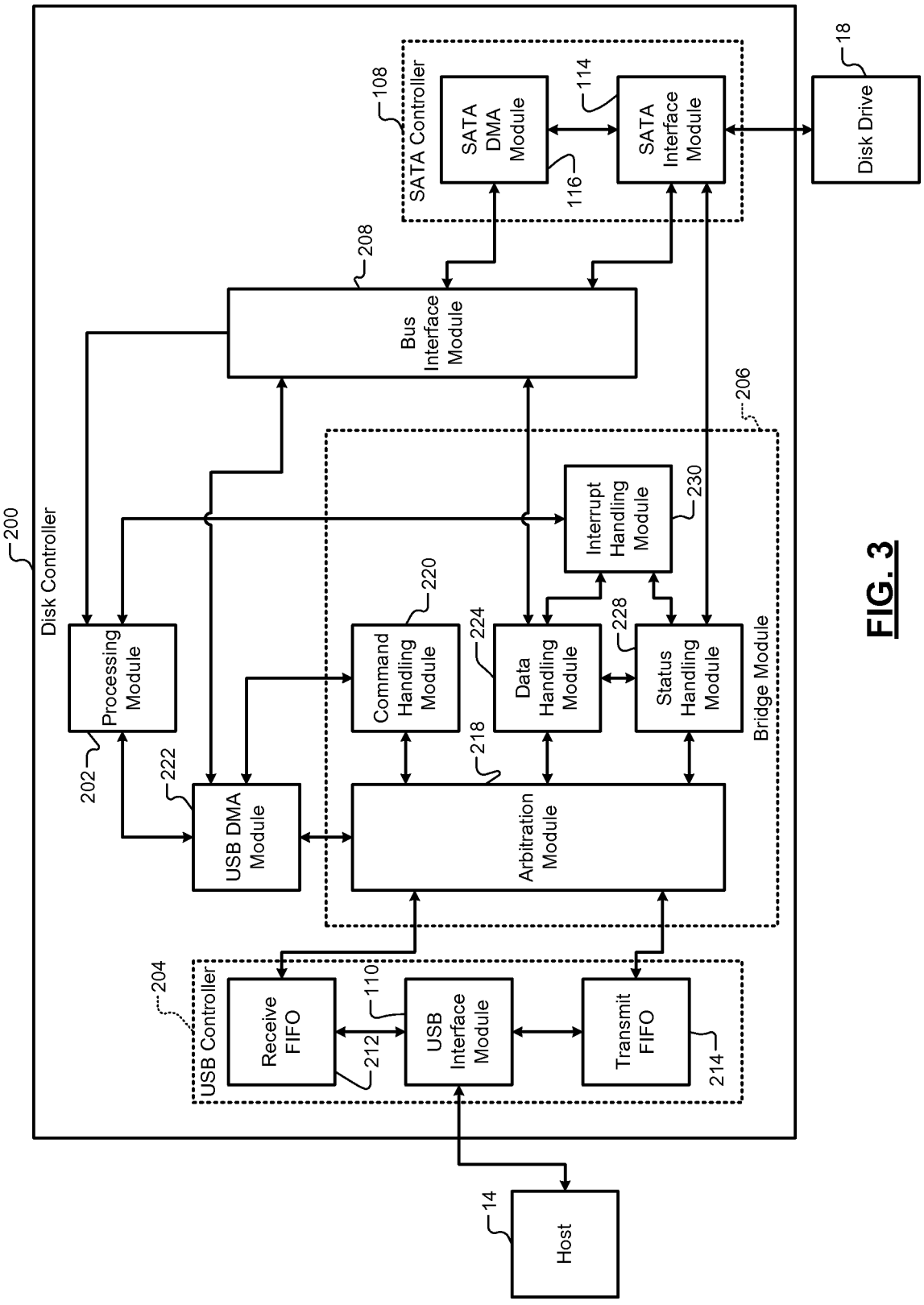
Disk Controller
200
Processing Module
202
USB DMA Module
222
Command Handling Module
220
Data Handling Module
224
Status Handling Module
228
Interrupt Handling Module
230
Bridge Module
206
Arbitration Module
218
Bus Interface Module
208
SATA Controller
108
SATA DMA Module
114
116
SATA Interface Module
Disk Drive
18
USB Controller
204
Receive FIFO
110
212
USB Interface Module
Transmit FIFO
214
Host
14

FIG. 3

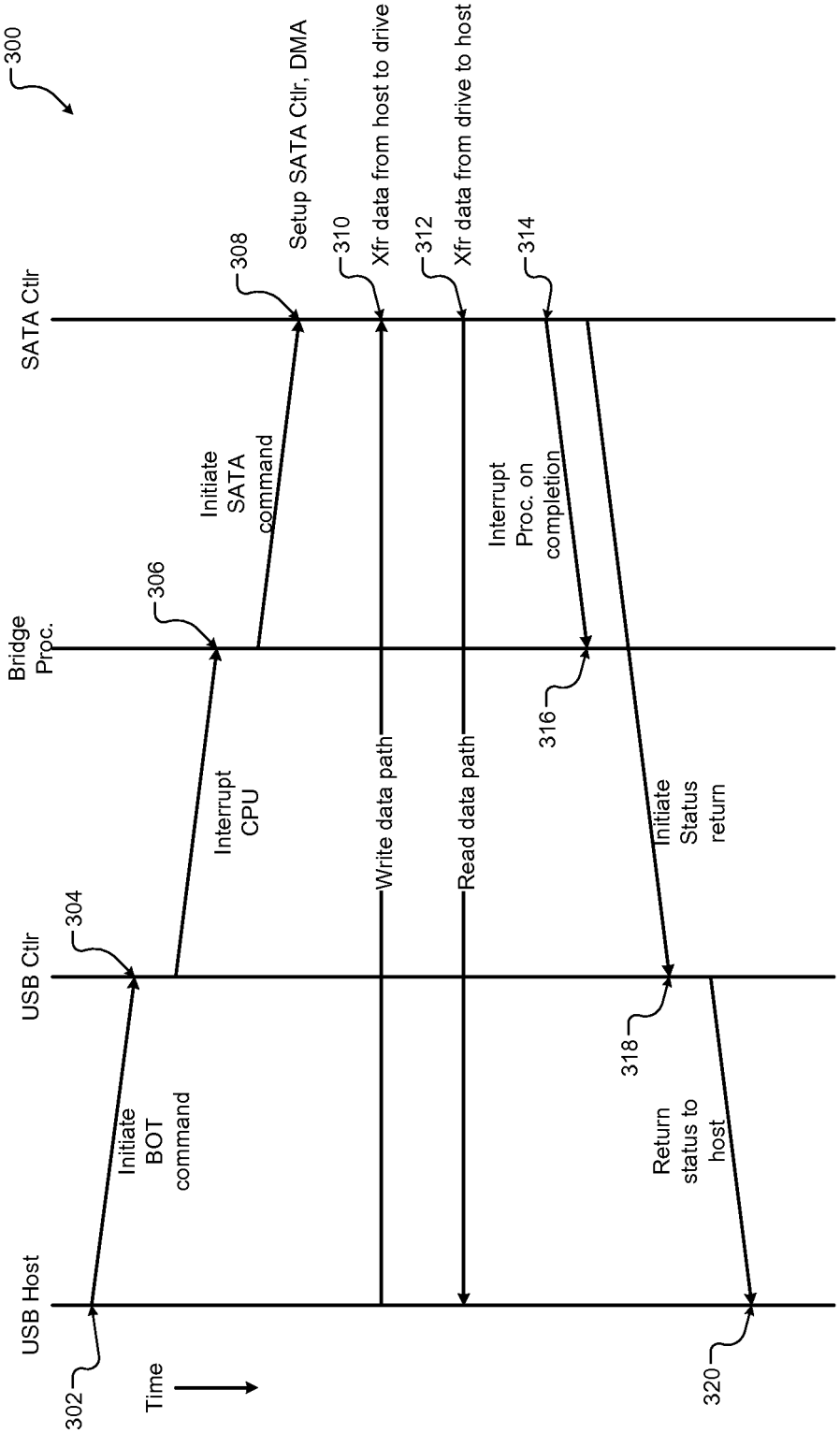
300
SATA Ctlr
Bridge Proc.
USB Ctlr
USB Host
Time
302
304
306
308
310
312
314
316
318
320
Initiate BOT command
Interrupt CPU
Initiate SATA command
Setup SATA Ctlr, DMA
Write data path
Xfr data from host to drive
Read data path
Xfr data from drive to host
Interrupt Proc. on completion
Initiate Status return
Return status to host

FIG. 4

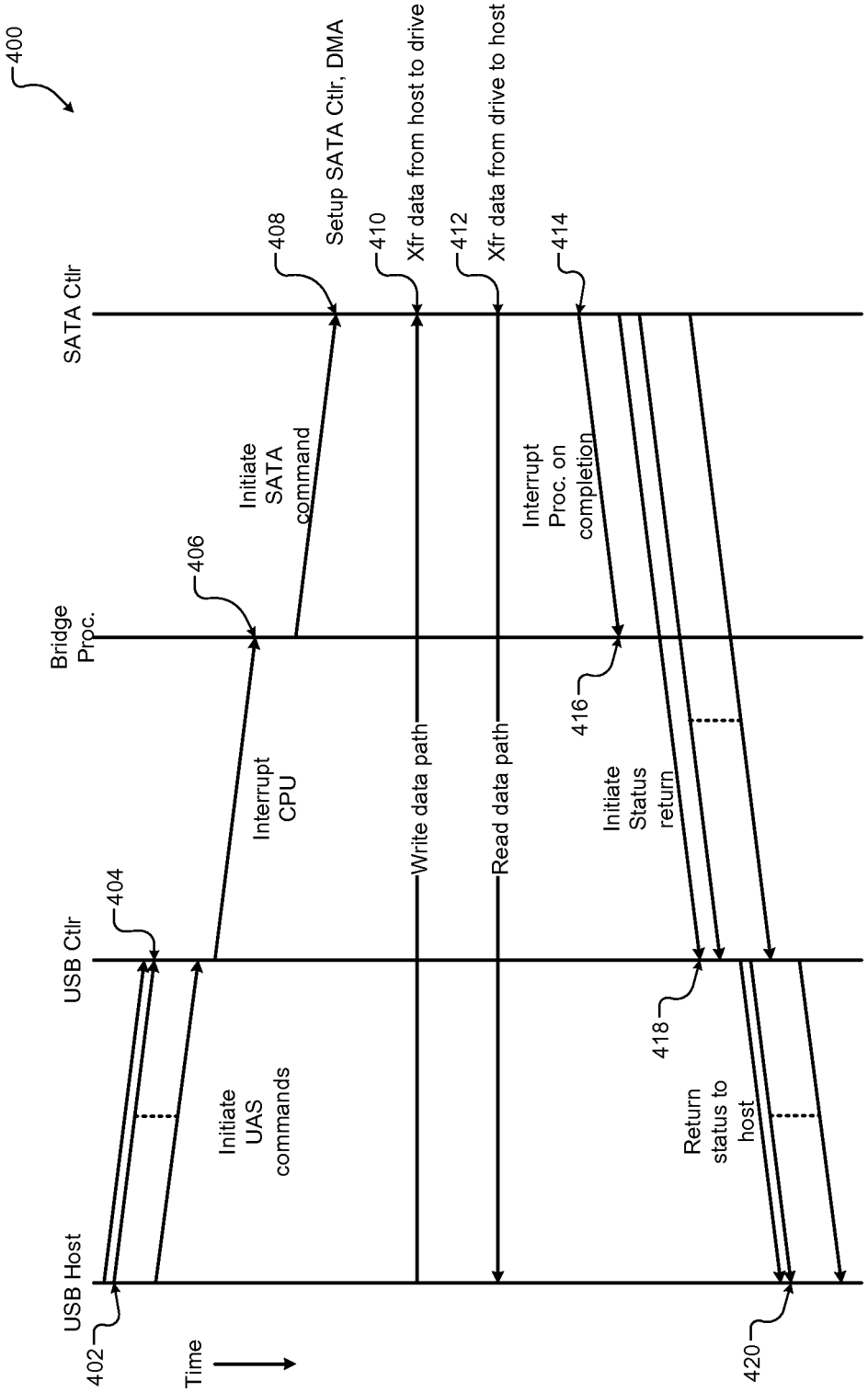
400
SATA Ctlr
Bridge Prpc.
USB Ctlr
USB Host
Time
402
404
406
408
410
412
414
416
418
420
Setup SATA Ctlr, DMA
Xfr data from host to drive
Xfr data from drive to host
Initiate SATA command
Interrupt Proc. on completion
Interrupt CPU
Write data path
Read data path
Initiate Status return
Initiate UAS commands
Return status to host

FIG. 5

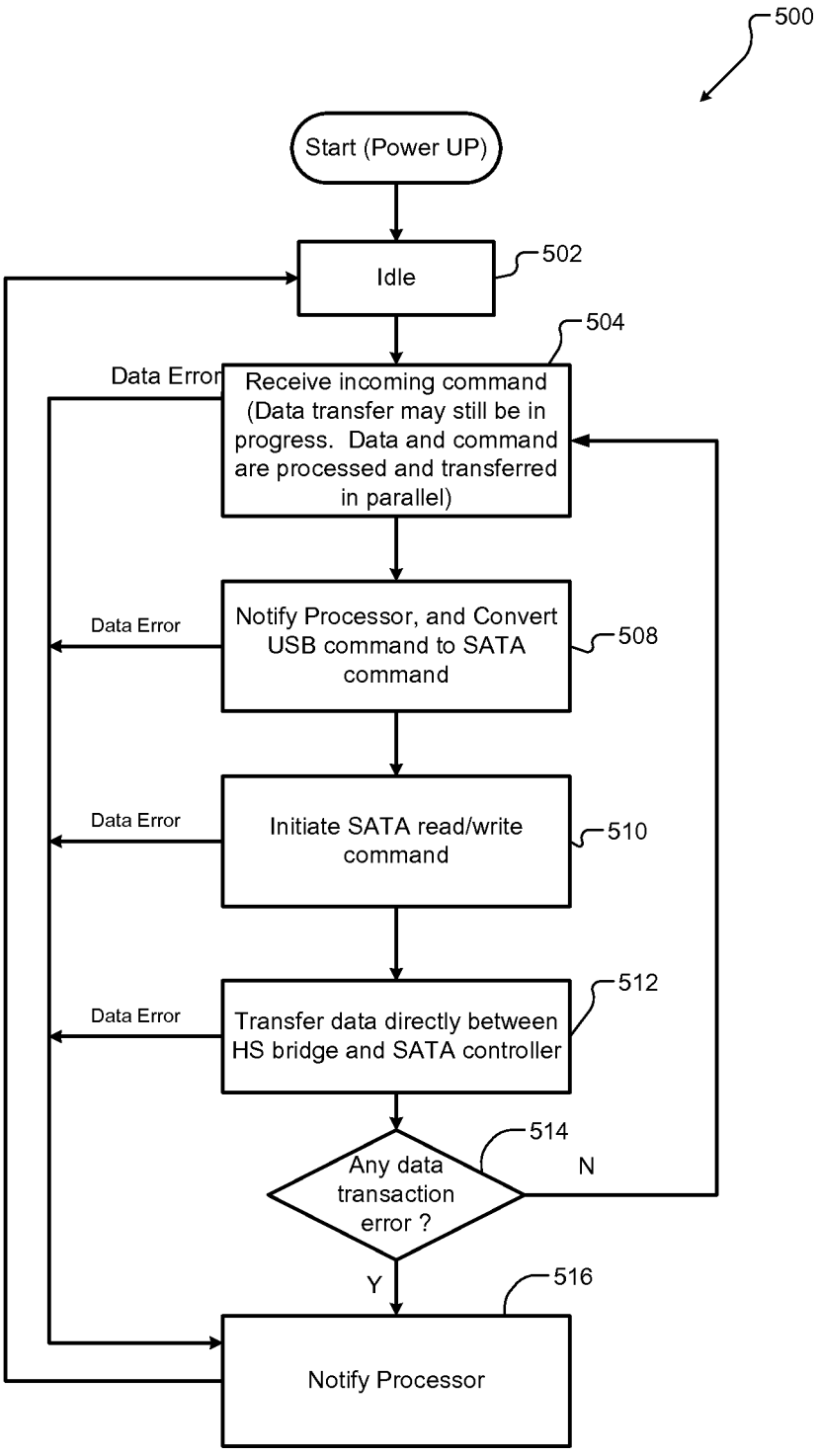
500
Start (Power UP)
Idle
502
504
Data Error
Receive incoming command (Data transfer may still be in progress. Data and command are processed and transferred in parallel)
Data Error
Notify Processor, and Convert USB command to SATA command
508
Data Error
Initiate SATA read/write command
510
Data Error
Transfer data directly between HS bridge and SATA controller
512
514
Any data transaction error ?
N
Y
516
Notify Processor

FIG. 6A

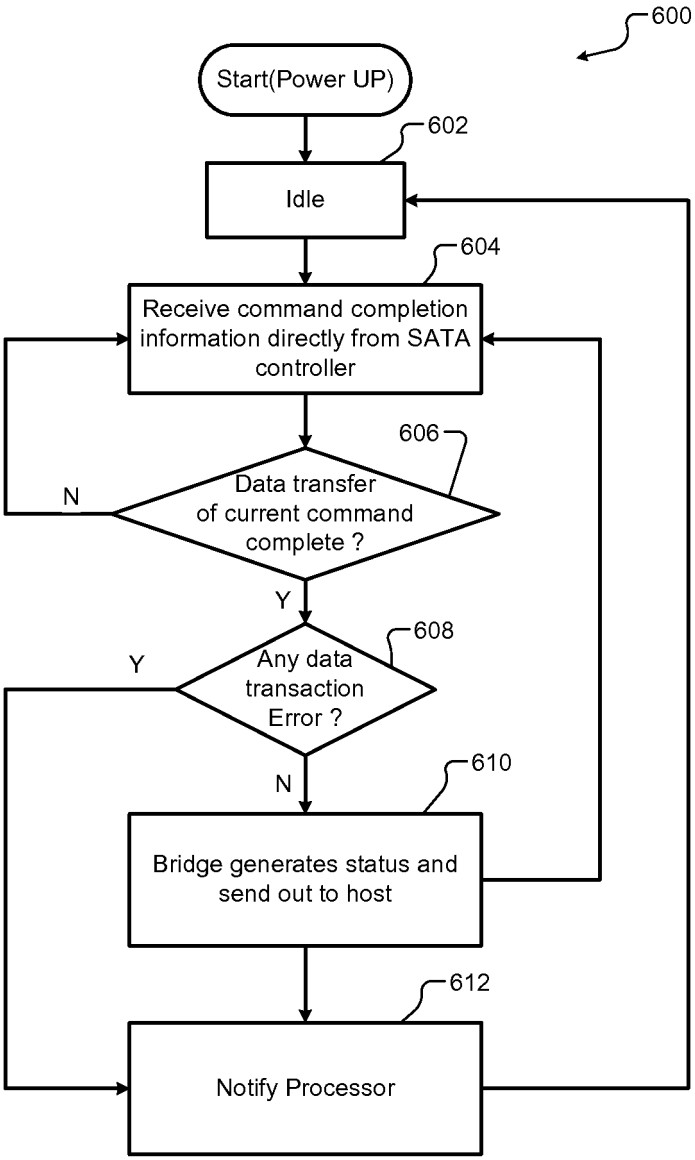
600
Start(Power UP)
602
Idle
604
Receive command completion information directly from SATA controller
606
Data transfer of current command complete ?
N
Y
608
Any data transaction Error ?
Y
N
610
Bridge generates status and send out to host
612
Notify Processor

FIG. 6B

REFERENCES CITED IN THE DESCRIPTION

This list of references cited by the applicant is for the reader's convenience only. It does not form part of the European patent document. Even though great care has been taken in compiling the references, errors or omissions cannot be excluded and the EPO disclaims all liability in this regard.

Patent documents cited in the description

- US 27637111 A **[0001]**
- US 61405444 A **[0001]**
- US 2010185808 A1 **[0006]**
- US 2007005838 A1 **[0007]**
- US 7620747 B1 **[0008]**